UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

TRABAJO FIN DE GRADO

CONTRIBUCIONES A UN SISTEMA DE DETECCIÓN DE PERSONAS EN CÁMARAS OMNIDIRECCIONALES

Autor: Javier Expósito Cáceres

Tutor: Pablo Carballeira López

Ponente: Jesús Bescós Cano

JULIO 2021

CONTRIBUCIONES A UN SISTEMA DE DETECCIÓN DE PERSONAS EN CÁMARAS OMNIDIRECCIONALES

Autor: Javier Expósito Cáceres
Tutor: Pablo Carballeira López
Ponente: Jesús Bescós Cano

Video Processing and Understanding Lab
Departamento de Tecnología Electrónica y de las Comunicaciones
Escuela Politécnica Superior
Universidad Autónoma de Madrid
JULIO 2021

Trabajo parcialmente financiado por el Gobierno de España bajo el proyecto TEC2017-88169-R (MobiNetVideo)

A mis padres…

Resumen

Actualmente la inteligencia artificial (IA) y el Deep Learning son unos de los campos con más proyección.

Las redes neuronales convolucionales (CNN) han abierto un abanico de posibilidades muy amplio en el ámbito de la visión computacional ya que han demostrado un gran rendimiento en tareas de clasificación detección y categorización de imágenes, mejorando exponencialmente el rendimiento y tiempos computacionales de los algoritmos ya existentes. Además, esta tecnología está cada vez más incluida en nuestro día a día, facilitando y realizando de forma autónoma tareas de nuestra vida cotidiana. Unos ejemplos donde esta tecnología está totalmente instaurada serían tareas como la conducción autónoma de coches, la detección de personas en videovigilancia o incluso en procesos de predicción de patrones.

Durante el desarrollo de este trabajo se ha realizado un estudio sobre cómo mejorar el rendimiento de un sistema de detección de personas en cámaras omnidireccionales.

El sistema de detección propuesto utiliza un modelo preentrenado de CNN, durante la etapa de extracción de características y utiliza un Grid de clasificadores repartidos sobre toda la superficie de la imagen durante la etapa de detección. Este Grid es capaz de adaptarse a la distorsión introducida por este tipo de cámaras.

Para mejorar el rendimiento del sistema se han aplicado técnicas de aumento de datos en las secuencias de entrenamiento. Por otra parte, también se amplió el número de clasificadores empleados en la etapa de detección.

Finalmente se realizó una comparación entre los distintos casos estudiados para evaluar el impacto de los cambios introducidos.

Palabras clave: Aprendizaje automático, redes neuronales convolucionales, aprendizaje profundo, Grid de clasificadores, detección de personas, cámaras omnidireccionales

Abstract

Currently, artificial intelligence (IA) and Deep Learning are one of the fields with the most future projection.

Convolutional neural networks (CNN) have opened a very wide range of possibilities in the field of computational vision since they showed great performance in classification, detection and categorization of images, exponentially improving the performance and computational times of algorithms that already exist. In addition, this technology is increasingly included in our day-to-day lifes, facilitating and autonomously performing our daily chores. Some examples in which this technology is fully established could be autonomous car driving, detection of people in video surveillance or pattern prediction processes.

During the development of this project, a study was carried out of how to improve the performance of a people detection system in omnidirectional cameras.

The proposed detection system uses a pre-trained model of CNN during the feature extraction stage and a Grid of classifiers spread over the entire image surface during the detection stage. This Grid is able to adapt to the distortion introduced by this type of camera.

To improve the performance of the system, data augmentation techniques have been applied in the training sequences. On the other hand, the number of classifiers used in the detection stage was also extended.

Finally, a comparison was made between the different studied situations to evaluate the impact of the introduced changes.

Key words: Machine Learning, convolutional neural networks, Deep Learning, Grid of classifiers, people detection, omnidirectional cameras

Agradecimientos

En primer lugar me gustaría dar las gracias a mi familia por todo apoyo y paciencia que tienen conmigo día a día. También a mis abuelos que siempre han estado pendientes de mi desarrollo académico.

Por otro lada a mi tutor Pablo, por estar siempre dispuesto ayudarme durante el desarrollo de este trabajo.

Para acabar me gustaría hacer una mención especial a todos y cada uno de mis compañeros de la facultad con los que he compartido mis aventuras: María, Carlos, Laura, Naran, Sergio, Tomas, Alfonso, Andrés y muchos mas que me dejo sin nombrar.

A mis amigos de Tres Cantos, los de toda la vida, Gabri, Miguel, Adri G, Xapu, Juan, Oliva, Adri R, Alex, Arturo, Alicia, Jose. Muchas gracias por crecer conmigo.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Motivación

En el ámbito de la visión computacional la detección de objetos en imágenes es uno de los métodos más estudiados y desarrollados. Con la llegada de las CNN se ha conseguido mejorar el rendimiento de los algoritmos ya existentes donde uno de los principales inconveniente era su coste computacional, esto limitaba su uso en aplicaciones en tiempo real. Esta mejora ha despertado un gran interés en el campo de la vídeo vigilancia [1].

Normalmente cuando hablamos de cámaras la primera idea que nos viene a la cabeza es el uso de una cámara convencional, este tipo de cámaras tienen un campo de visión muy limitado y las distorsiones introducías son prácticamente inapreciables. Existe otro tipo de cámaras, llamadas cámaras omnidireccionales que aumentan el campo de visión lo que supone una gran ventaja en tareas de videovigilancia ya que permite por ejemplo vigilar una sola instancia con una sola cámara reduciendo así la cantidad de datos generados por un sistema multicámara. A costa de aumentar el campo de visión como consecuencia aumentan también las distorsiones introducidas. Estas distorsiones son totalmente irregulares, siendo mucho más apreciables en el borde de la imagen.

Existen multitud de algoritmos basados en CNN que ofrecen gran rendimiento en imágenes convencionales, pero no logran adaptarse a las deformaciones de introducidas por las cámaras omnidireccionales. En [2] se realizó un sistema detector de personas en cámaras omnidireccionales que empleaba algoritmos clásicos de ventanas de deslizantes para extraer las características, más adelante en [3] se adapto para extraer las características mediante algoritmos basados CNN manteniendo el Grid de clasificadores espaciales.

La idea principal de este trabajo es estudiar como se puede mejorar el rendimiento del sistema desarrollado en [3] con el fin de aplicarlos a otros sistemas de detección.

1.2. Objetivos

El objetivo de este trabajo es llevar a cabo un estudio sobre cómo podemos mejorar el rendimiento de un sistema de detección de personas que utiliza técnicas de aprendizaje profundo en imágenes captadas con cámaras omnidireccionales. Para desarrollar este trabajo nos hemos centrado en los siguientes puntos:

- Añadir nuevas secuencias de test y entrenamiento desde otras bases de datos.
- Aplicado técnicas de aumento de datos para el entrenamiento.
- Aumentar el número de clasificadores en la etapa de detección.

Nuestro punto de partida es el trabajo de fin de máster [3] de un estudiante de esta universidad, que adapto un sistema de detección que empleaba métodos de ventanas deslizantes a métodos basados en redes neuronales convolucionales (CNN)[4]. En trabajos anteriores se obtuvieron buenos resultados para las distintas secuencias de vídeo que se pueden encontrar dentro de la base de datos conocida como PIROPO [5] (*People in Indoor Rooms with Perspective and Omnidirectional cameras*).

1.3. Organización de la memoria

Este documento está dividido en cinco capítulos. El primero está compuesto por un resumen global del trabajo y una explicación motivacional sobre las razones por la cuales se ha decidido llevarlo a cabo.

El segundo capítulo es el estado del arte. En este capítulo se ofrece una explicación detallada sobre el momento actual en el que se encuentran las tecnologías implicadas en el trabajo.

Durante el tercer capítulo se habla más profundamente sobre cómo se ha llevado a cabo la investigación, profundizando en detalles más técnicos de todos los conceptos y métodos aplicados.

En el cuarto capítulo se analizan los resultados obtenidos en cada uno de los casos estudiados.

En el quinto y último capítulo se realiza una pequeña reflexión de las soluciones aplicadas y, además, se proponen vías de estudio en el caso de que otro estudiante de esta u otra universidad desee continuar con la investigación.

Capítulo 2

Estado del arte

Este proyecto consiste en el desarrollo de un estudio del rendimiento de un sistema de detección de personas sobre imágenes capturadas por cámaras omnidireccionales.

La detección de personas es uno de los campos más estudiados en el ámbito de la visión computacional. En estos últimos años se han desarrollado algoritmos buscando aumentar la precisión y fiabilidad de los ya existentes. En este capítulo se expondrán los algoritmos más utilizados, sus ventajas e inconvenientes. Además, se expondrán las principales diferencias entre imágenes captadas por cámaras convencionales e imágenes captadas por cámaras omnidireccionales. Conocer las principales características de las imágenes omnidireccionales nos será útil para comprender por que estos métodos no se adaptan bien a este tipo de imágenes. Para finalizar, se detalla como se han adaptado los citados algoritmos para solucionar los problemas encontrados.

2.1. Algoritmos de detección de personas

Los algoritmos de detección ofrecen un gran rendimiento en la actualidad [6]. No obstante, se sigue buscando la manera de mejorar y crear nuevos algoritmos con los que se obtengan mejores resultados que los que ofrecen los ya existentes.

Como ya se ha comentado en la introducción de este trabajo, es un campo que en la actualidad está totalmente implantado en actividades de nuestro día a día, como pueden ser en tareas de videovigilancia o situaciones tan importantes como la conducción autónoma.

La gran dificultad de la detección de personas no es su coste computacional, sino que debe ser capaz de adaptarse a diversos escenarios y a las distintas características que existen entre las personas.

Para que un sistema de detección de personas funcione correctamente, debemos dotar al sistema de la capacidad de adaptarse a las siguientes situaciones:

- **Diferencias físicas entre personas**: no existen dos personas iguales.
- **Cambios posturales**: las personas no son objetos de formas rígidas, sino que tienen su propia fisiología que les permite adoptar diferentes posturas.
- **Diferencias en las condiciones del entorno**: son importantes la iluminación o la perspectiva.

2.1.1. Algoritmos clásicos

El algoritmo más utilizado para afrontar este tipo de tareas es el algoritmo de ventanas deslizantes[7]. Este algoritmo consta de dos pasos: en el primero, se realiza una extracción de las características más relevantes de la imagen y en el segundo, se realiza una clasificación de las características extraídas.

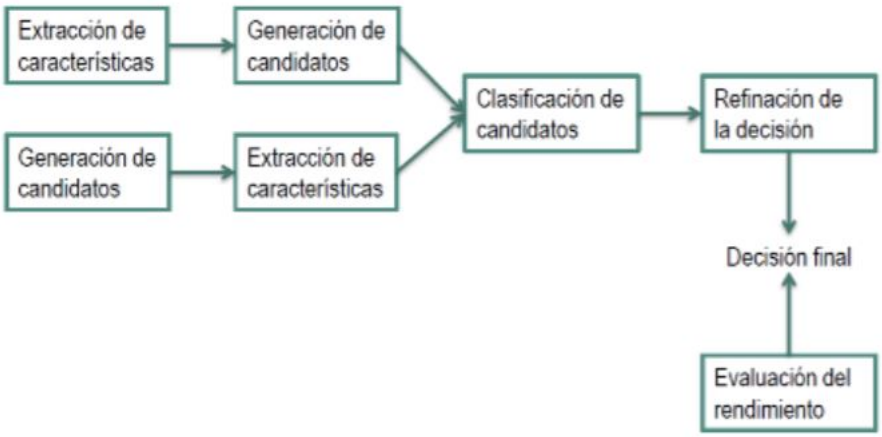

FIGURA 2.1: Esquema típico de las distintas etapas de un detector de objetos. Extraído [7].

Algoritmos de ventanas deslizantes

La base de este algoritmo de ventanas deslizantes [8], como bien indica su nombre, consiste en deslizar una ventana de tamaño de N x M píxeles sobre la imagen original. El recorrido que realiza la ventana normalmente es de izquierda a derecha de la imagen y de arriba a abajo. Es necesario que durante el desplazamiento se produzca un solapamiento grande entre ventanas.

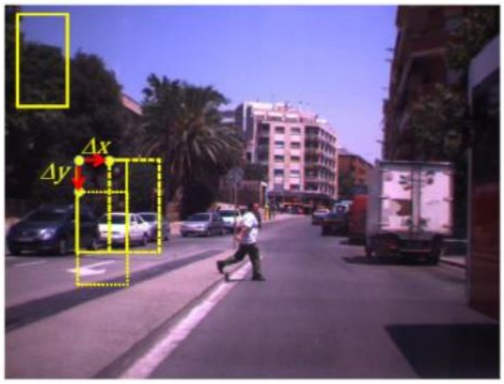

FIGURA 2.2: Ejemplo de desplazamiento de la ventana sobre la imagen, al aplicar ventanas deslizantes. Extraída de [7].

Por cada posición de la ventana se calcula un vector descriptor que contiene toda la información extraída de esa región de la imagen. Relacionando la finalidad del proyecto a este concepto, lo que buscamos en cada descriptor sería detectar la presencia o no de una persona.

Debido a la forma en la que capturamos el mundo, la posición de la persona respecto a la cámara influye enormemente en su apariencia en la imagen. Dos personas a distintas distancias respecto de la cámara, pueden tener la misma apariencia pero el tamaño de la persona más que se encuentra más lejos será mucho menor que la que se encuentra más cerca de la cámara. Para solucionar estos problemas de escala, se aplica el concepto de pirámide multiescalar [9]. Este concepto nos permite utilizar el método de ventanas deslizantes sobre las distintas versiones escaladas de la imagen original. Si se mantiene el tamaño de la ventana constante y se realiza el barrido sobre las distintas versiones escaladas, podremos detectar personas que durante el primer barrido habían pasado desapercibidas debido a su tamaño.

Los descriptores obtenidos, pasarán a ser los datos de entrada de un sistema de clasificación entrenado para determinar la presencia o ausencia de una persona. Una vez clasificados, debemos eliminar las detecciones redundantes y quedarnos únicamente con las de mayor valor. Este proceso es conocido como supresión de no máximos (NMS por sus siglas en inglés).

La principal desventaja de este algoritmo es su coste computacional, debido a que la resolución de la imagen y el propio tamaño de la ventana hacen que su coste aumente, limitando su uso para aplicaciones en tiempo real.

Extracción de características

Al contrario que las personas, los ordenadores no pueden identificar rápidamente la presencia de una persona en una imagen. Al procesar una determinada imagen, lo que el ordenador percibe es una matriz con un valor numérico para cada píxel. Mediante

los algoritmos de detección los ordenadores son capaces de extraer un vector de valores que represente la información de una manera interpretable por ellos. Para extraer estos vectores, existen diversos métodos:

Descriptores de gradientes orientados

Este método de extracción de características es también conocido en el ámbito de la detección de objetos como HOG [10] (por sus siglas en ingles **histogram of oriented gradients**). Este algoritmo se implementó en torno al año 2006. Tras años de investigación, se han conseguido tan buenos resultados que en la actualidad existen aplicaciones en tiempo real que hacen uso de estos descriptores.

Los descriptores HOG son uno de los posibles métodos empleados por el algoritmo de ventanas deslizantes para extraer las características. Para cada posición de la ventana se calcula un descriptor que se corresponde con un histograma que representa la dirección del gradiente en función de cómo varía el valor de cada píxel respecto a sus píxeles vecinos. Los valores altos del histograma corresponden con los principales gradientes de cada dirección.

El vector de características calculado será una concatenación de todos los histogramas de la imagen, que contienen todas las características que la componen. Este vector será la entrada de un clasificador que determinará si existe la presencia de personas.

Uno de los beneficios de utilizar este método de extracción de características es que dota al sistema con la capacidad de ser independiente a cambios de iluminación de la escena, ya que lo que se calcula es la dirección en que varía la intensidad de cada píxel respecto a sus píxeles vecinos, logrando que la representación de una persona sea igual en distintas condiciones de iluminación.

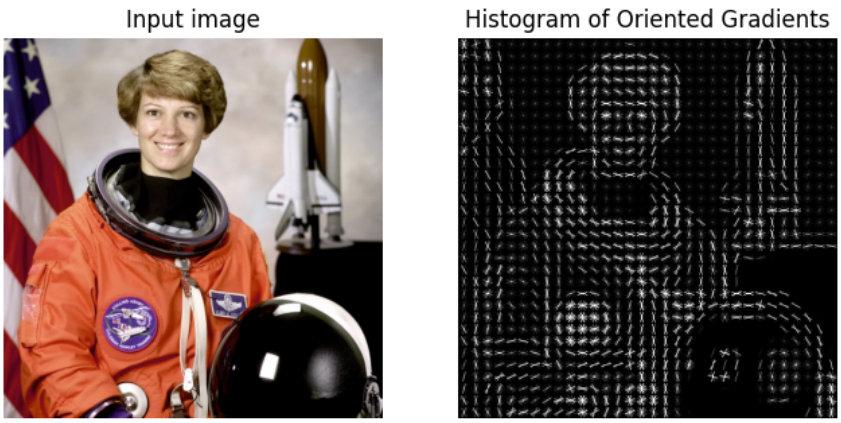

FIGURA 2.3: Ejemplo de extracción de características mediante descriptores HOG.

Características HAAR

Al igual que HOG, los descriptores HAAR [11] son un método de extracción de características de una imagen, para cada posición de la ventana se realiza un filtrado de la región, utilizando unos filtros llamados filtros de HAAR básicos, como los representados en la Figura 2.4. El resultado de este filtrado nos permite percibir los cambios de intensidad en cualquier posición de la imagen. Por ejemplo, si aplicamos este método sobre la imagen de una persona, acabaríamos obteniendo como resultado su contorno.

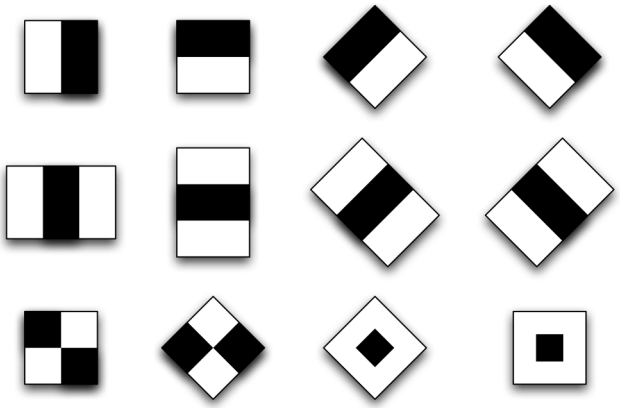

FIGURA 2.4: Filtros básicos HAAR.

El descriptor final será la concatenación de todos los resultados obtenidos tras el filtrado. Como se puede imaginar, el coste de este algoritmo es alto y depende directamente del tamaño de la ventana y la resolución de la propia imagen. Si utilizamos una ventana grande, la precisión de los descriptores obtenidos será insuficiente. Si, por el contrario, utilizamos una ventana demasiado pequeña, el coste computacional se elevará demasiado.

Clasificadores basados en aprendizaje automático

Los clasificadores se encargan de analizar los descriptores obtenidos en la etapa de extracción de características de una imagen. Su función es determinar si existe o no la presencia de un determinado objeto, en el caso de este proyecto una persona.

Los clasificadores basados en aprendizaje automático necesitan un entrenamiento previo, donde el clasificador aprende a diferenciar la presencia o no de una persona.

Uno de los clasificadores con más importancia en los últimos años y más utilizado es la máquina de soporte vectorial (SVM) [12] actualmente está siendo desplazado por algoritmos basados en redes neuronales.

Máquina de soporte vectorial

Es un algoritmo que emplea el aprendizaje automático, pensado para resolver problemas de clasificación y regresión. Originalmente fue diseñado para casos de clasificación binaria, pero tras años de investigación, se ha conseguido un gran rendimiento para un abanico de situaciones muy diversas, siendo un referente en el Machine Learning.

El concepto principal de la máquina de SVM [13] es el hiperplano, representado en la Figura 2.5. Lo que se busca con el uso de este algoritmo es calcular un plano que divida los datos de entrada entre las diferentes clases minimizando la función de costes.

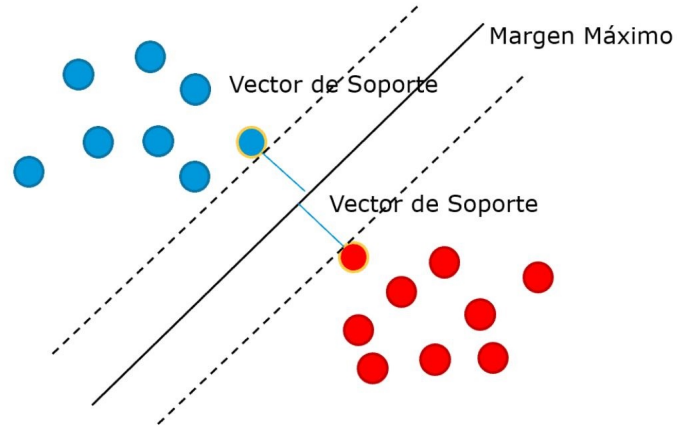

FIGURA 2.5: Ejemplo de SVM de dos clases en dos dimensiones.

El hiperplano debe de calcularse de forma que separe los datos de entrada entre las posibles clases, en la detección de personas solo existen dos clases: Una clase que indica la presencia de una persona y otra que indica el caso contrario, es decir, no se detecta persona.

Estas máquinas permiten al usuario controlar el rendimiento del clasificador variando dos parámetros:

- Parámetro de regulación o parámetro C. Este parámetro permite flexibilizar la rigurosidad del clasificador ante los fallos permitiendo así controlar el sobreentrenamiento del clasificador.
- Margen de separación o parámetro D. Este parámetro permite controlar la prioridad de una clase frente a otra permitiendo ajustar así la cantidad de falsos positivos o falsos negativos.

2.1.2. Detectores basados en CNN

Con la llegada de las redes neuronales [4], se han desarrollado algoritmos más avanzados que obtienen mayor rendimiento que los algoritmos clásicos.

Para la detección de objetos se utilizan redes neuronales convolucionales [14], también conocidas como CNN. Este tipo de redes son una modificación de las redes neuronales convencionales.

Para comprender modelos de redes neuronales complejos es importante conocer la unidad básica llamada neurona o también conocida como perceptrón [**neu**].

La neurona recibe una serie de datos de entrada $(\vec{x})$ y genera una salida $(\vec{y})$ que se puede expresar mediante una combinación lineal entre la entrada y el peso de la neurona $(\vec{w})$. Normalmente a la salida se le aplica una función no lineal, llamada función de activación [15].

En la Figura 2.6 podemos observar un esquema gráfico en el que se puede ver la similitud de los perceptrones a las neuronas biológicas de las personas.

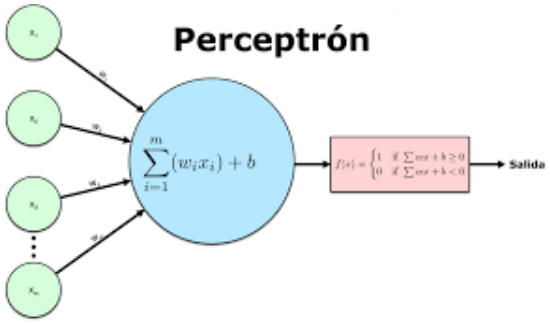

FIGURA 2.6: Esquema de la arquitectura de un perceptrón de una red neuronal.

Redes neuronales convolucionales

Este tipo de redes están diseñadas específicamente para el tratamiento de imágenes. En esta sección vamos a ver sus principales características.

Estas redes están formadas por varias capas y cada una de ellas tiene una función determinada. Esto nos permite que las redes extraigan las características de cada imagen y realicen tareas de detección, categorización de objetos o clasificación de escenas.

Diseñar redes neuronales convolucionales no es algo trivial, por lo que normalmente se utilizan modelos ya preentrenados que han demostrado tener una gran capacidad de aprendizaje y rendimiento en reconocimiento de imagen.

En este tipo de redes, podemos destacar distintos tipos de capas:

Capas convolucionales

Estas capas se caracterizan por aplicar una operación de convolución [16] sobre la

imagen, utilizando K filtros distintos. Como consecuencia, para cada imagen se obtendrán K salidas.

La finalidad de estas capas es extraer características propias de la imagen. A la salida de estas capas se suele utilizar la función de activación tipo ReLU, eliminando los valores negativos y conservando los positivos se consigue que las CNN implementen funciones no lineales.

Como los datos de salida de esta capa son muy elevados, normalmente lo que se encuentra inmediatamente después es una capa de pooling.

Capas pooling

La finalidad de esta capa es reducir las dimensiones de salida de las capas convolucionales. Aunque existen diversos métodos, el más empleado es el de Max-pooling [17]. En este método para cada una de las salidas obtenidas en la capa anterior, primero se realiza una división en bloques de N x M píxeles, para posteriormente seleccionar el valor más alto y representativo de cada bloque.

Esto nos permite reducir considerablemente el volumen de datos y, como consecuencia, el número de neuronas de nuestra red.

Capas fully connected

Esta es la última capa de las redes convolucionales, la capa clasificadora. Tendrá tantas salidas como clases a identificar [18].

Algoritmos basados en CNN

Dentro de los detectores que utilizan redes convolucionales [19] podemos clasificarlos en dos grupos:

- **Detectores con segmentación**

  Estos detectores se caracterizan por dividir el proceso de detección en dos etapas diferentes. Primero llevan a cabo una división de cada una de las imágenes de entrada en regiones, conocidas como RoI, en las cuales existe una mayor probabilidad de producirse una detección. Cada región es enviada por separado a una CNN la cual realiza la clasificación.

  - **R-CNN [20]:** Es un ejemplo de redes que emplean esta metodología. El principal inconveniente es su alto coste computacional, dado que el tiempo medio por imagen es de 47 segundos. Este modelo, mediante la búsqueda selectiva, genera 2000 RoI por imagen, estas regiones son tratadas por una CNN que

extrae sus características. La salida generada por la red es un vector características, que es la entrada de un SVM que devuelve la probabilidad de pertenecer a una clase.

- **Fast RCNN [21]:** Se trata de una modificación del esquema propuesto para las R-CNN, con el objetivo de mejorar el tiempo de procesamiento. En estas redes se suprime el uso del algoritmo de búsqueda selectiva para determinar las RoI. Como consecuencia la entrada de la CNN serán las imágenes y como salida se obtendrá un mapa de características convolucional. Mediante un algoritmo de búsqueda aplicado sobre los mapas de características obtenidos anteriormente, se localizarán las regiones de interés que serán redimensionadas para ser tratadas como entrada de una última capa fully connected, que genera como salida los vectores característicos de las RoI propuestas. Finalmente, estos vectores se clasificarán en una capa de softmax para extraer la probabilidad de pertenecer a una clase. Debido a las modificaciones realizadas, se obtiene una mejora en el tiempo de computación aproximadamente 140 veces menor que en los modelos anteriores.
- **Faster RCNN [21]:**Este nuevo esquema busca mejorar el rendimiento de los modelos anteriores, la idea es la misma, pero se elimina el algoritmo de búsqueda selectiva y se añade una red neuronal adicional para extraer las RoI. El tiempo medio de este tipo de redes es de 0,2 segundos por imagen.

- **Detectores de un solo paso:** Estos detectores realizan la extracción de características y la clasificación al mismo tiempo, priorizando la velocidad de computación para posibilitar su uso en aplicaciones en tiempo real.

  - **YOLO (You Only LOOK Once) [22]:** Esta arquitectura utiliza solamente una CNN para todo el proceso. Comienza dividiendo la imagen de entrada en regiones de SxS píxeles y para cada región se define M Bounding Box. Cada una de las Bounding Box propuestas son tratadas por la CNN que determina la probabilidad de pertenecer a una clase. Si esa probabilidad supera el umbral la CNN detecta y localiza el objeto.
  - **Single Shot MultiBox Detector (SSD) [23]:** Esta arquitectura, usa una CNN para extraer las características de la imagen, pero sustituye la capa fully connected por capas convolucionales extras. Esto mejora la detección de objetos pequeños dentro de la imagen. La arquitectura SSD utiliza capas profundas de la red para la detección y clasificación de los objetos.

2.2. Cámaras omnidireccionales

En tareas como la videovigilancia, utilizar cámaras omnidireccionales frente a cámaras convencionales supone una gran ventaja, dado que estas tienen un mayor campo

de visión y permiten capturar una estancia completa solamente utilizando una sola cámara. Esto nos permite eliminar la dependencia de sistemas multicámara reduciendo el presupuesto y la cantidad de datos generados del sistema a implementar.

Las cámaras omnidireccionales [24] ofrecen un campo de visión mucho mayor que las convencionales, existen modelos de cámaras omnidireccionales con un rango de visión de 360º. Las más comunes y empleadas son las cámaras de ojo de pez.

2.2.1. Imágenes omnidireccionales

Existe una gran diferencia entre una imagen tomada por una cámara convencional frente a otra capturada con una cámara omnidireccional y es importante comprender los problemas a los que nos vamos a enfrentar.

Aunque en las imágenes de cámaras convencionales se introducen pequeñas deformaciones, en la Figura 2.7 se puede apreciar que estas deformaciones se acentúan en el caso de las imágenes omnidireccionales.

FIGURA 2.7: En esta imagen se pueden apreciar las deformaciones introducidas por las cámaras omnidireccionales.

Dentro de las propias imágenes omnidireccionales, dependiendo de donde se situé la persona respecto a la cámara, sufrirá más o menos distorsión.

Estratégicamente este tipo de cámaras se colocan en lugares elevados, como techos y paredes, para aprovechar en totalidad su campo de visión. Esto implica que las distorsiones introducidas por este tipo de cámaras afecten de la siguiente manera. Si la persona se sitúa por el centro de la imagen, no sufrirá gran deformación geométrica, pero su tamaño aumentará considerablemente respecto al resto de la escena. Sin embargo, si una persona se sitúa al borde de la escena, sufrirá grandes deformaciones geométricas, reduciendo su tamaño respecto a los objetos en la zona central.

Las deformaciones introducidas por este tipo de cámaras dificultan las tareas de detección de objetos en las imágenes capturadas. Por ejemplo, si los descriptores y clasificadores están adaptados a imágenes convencionales que no introducen grandes deformaciones, al aplicarlos sobre imágenes omnidireccionales no se adaptarán de manera correcta.

2.2.2. Solución a deformaciones introducidas por las cámaras omnidireccionales

Los métodos de detección de imágenes comentados anteriormente asumen que las personas mantienen grandes parecidos con su apariencia original. En las imágenes omnidireccionales, esto no es cierto, puesto que sufren grandes deformaciones dependiendo de su posición.

Existen diferentes métodos para adaptarse a estos problemas; uno de los más utilizados consiste en transformar las imágenes omnidireccionales en imágenes convencionales. Aun así, el problema no queda resuelto dado que durante esta transformación se pierde gran cantidad de información o incluso hay casos en los que las deformaciones se ven incrementadas, como se puede apreciar en la Figura 2.8

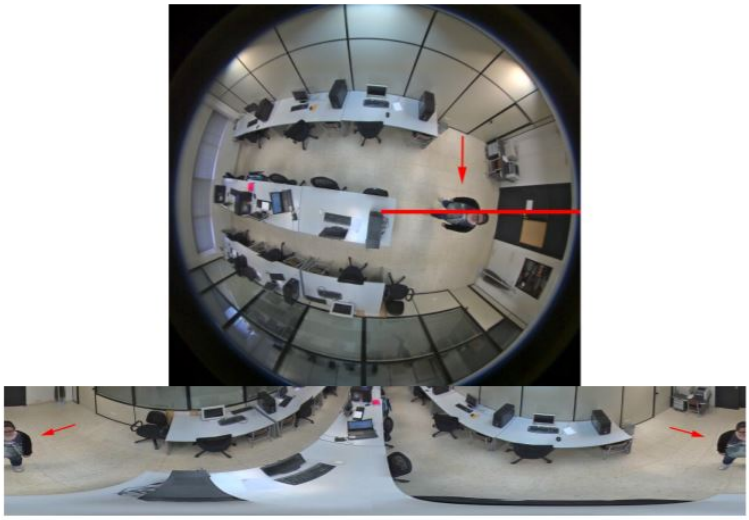

FIGURA 2.8: Imagen del resultado de transformar una imagen omnidireccional a imagen tradicional, extraída de [3]

Para llevar a cabo esta transformación, se realiza un muestreo de la imagen proyectada sobre un cilindro o una esfera, proyectando los puntos sobre una imagen plana. De esta manera, se consigue adaptar la imagen a los algoritmos convencionales.

Otra solución es la que se propone en [25] y [26], que es adaptar los filtros empleados en las capas convolucionales de la red para suplir las deformaciones de estas imágenes.

2.3. Posibles arquitecturas

En esta sección detallaremos brevemente las arquitecturas empleadas tradicionalmente en los detectores de personas en imágenes omnidireccionales, incidiendo en sus distintas etapas.

2.3.1. Grid of Spatial-Aware Classifiers

Este clasificador *Grid of Spatial-Aware Classifiers* (GSAC) [2], se caracteriza por emplear un Grid de clasificadores SVM repartidos por toda la imagen y un método de extracción de características como los comentados en la Sección 2.1.1 para obtener el vector características. Este clasificador al utilizar un único vector como descriptor global de la imagen es capaz de utilizarse en aplicaciones en tiempo real.

Este vector características que será la entrada del Grid de clasificadores previamente entrenados para detectar la presencia de una persona en su área de detección conocida como Fóvea. Como hemos visto en la Sección 2.2.1 las distorsiones introducidas por las cámaras omnidireccionales varían con la ubicación de la persona, utilizando este tipo de clasificador solucionamos este problema. Cada clasificador aprende la apariencia de una persona sobre su área de detección de manera individual , esa apariencia será distinta en función de su posición sobre la escena.

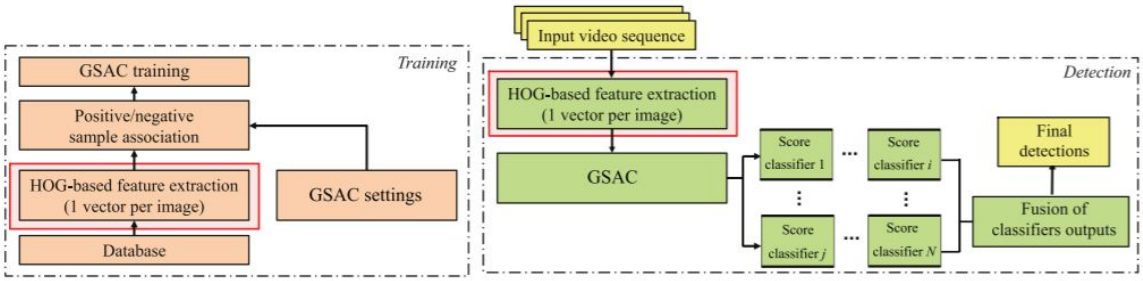

FIGURA 2.9: Diagrama de bloques de GSCA, extraída de [2]. Donde podemos observar que mediante un proceso de extracción de características se obtiene el vector descriptor será la entrada del Grid de clasificadores

En la Figura 2.9, observamos como este detector consta de dos etapas: En la primera etapa, la etapa de entrenamiento, los clasificadores aprenden la apariencia de una persona sobre su región a detectar. En la segunda, la etapa de clasificación, emplea los descriptores ya entrenados para detectar la presencia de personas en la imagen.

2.3.2. Deep Learning GSAC

Los detectores Deep Learning GSAC [3] siguen el mismo diagrama de bloques que el detector descrito en la Sección 2.3.1 , pero utiliza una CNN para extraer y clasificar las características de la imagen.

En el la Sección 3.1 del capitulo 3 se ofrece una descripción completa de este detector, ya que es el que hemos usado durante toda el trabajo.

Capítulo 3

Desarrollo e implementación

La idea de este trabajo de fin de grado es mejorar el rendimiento de un sistema de detección de personas para cámaras omnidireccionales. En este capítulo se van a exponer las características del sistema empleado y los principales problemas a los que nos hemos enfrentado junto con soluciones aplicadas.

3.1. Sistema propuesto

Partimos de un sistema de detección de personas en cámaras omnidireccionales desarrollado por Lorena García en [2], el sistema original empleaba un método de ventanas deslizantes para extraer las características y en la etapa de clasificación utilizaba una SVM. Más adelante, en [3], este mismo sistema se modificó para realizar la etapa de extracción y clasificación mediante una CNN.

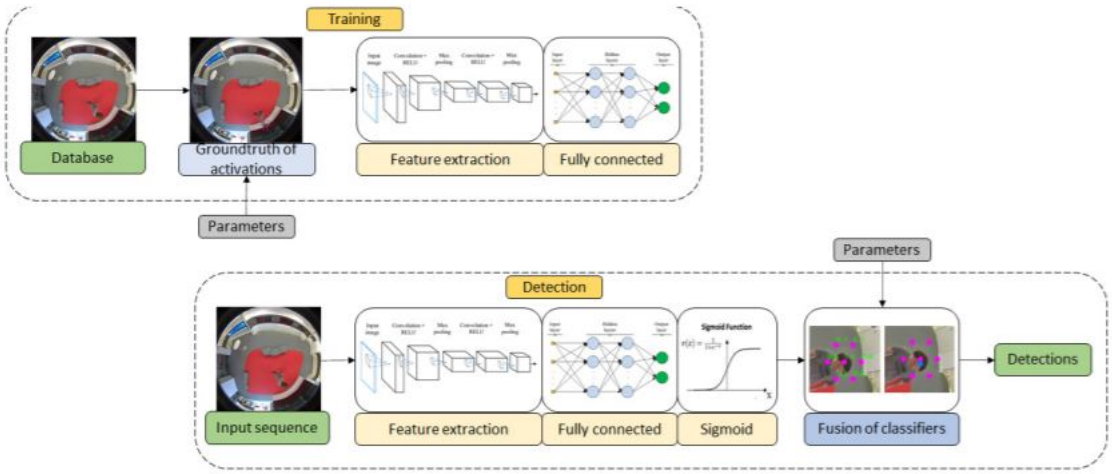

FIGURA 3.1: Diagrama de bloques del detector basado en CNN. Extraído de [3]. Este diagrama representa las dos etapas del sistema de detección, el diagrama superior corresponde con la etapa de entrenamiento, el diagrama inferior representa la etapa de detección.

EL sistema propuesto, como se puede ver en la Figura 3.1, consta de dos etapas. Una primera etapa de entrenamiento que utiliza el diagrama de bloques superior y posteriormente una segunda etapa de detección representado en el diagrama inferior.

El objetivo es entrenar una red neuronal que extraiga las características y realice la clasificación. Concretamente la red neuronal utilizada es una CNN preentrenada, Resnet18 [27], la razón para utilizar modelos preentrenados es que al especializar el modelo a nuestro dataset el coste computacional del entrenamiento se reduce y además ofrece mejores resultados a la hora de generalizar.

Durante la etapa de detección, la CNN se encarga de extraer las características de las imágenes de entrada y genera como salida un vector de predicciones que será la entrada del Grid de clasificadores. El vector de predicciones contiene la probabilidad de que se haya producido la detección de una persona en una determinada zona de la imagen. Para calcular esta probabilidad, aplicamos a la salida de la capa fully conected una función de activación tipo sigmoide, que limita los valores del vector entre 0 y 1. El vector de predicciones cuenta con N valores, siendo N el número de clasificadores del Grid del sistema. Finalmente, este vector será la entrada de un Grid de clasificadores que determinará la posición final de la persona. Esto se puede apreciar en el diagrama de bloques inferior de la Figura 3.1, correspondiente con la etapa de detección.

Estos clasificadores se reparten estratégicamente formando un Grid por la superficie de toda la imagen, pudiendo adoptar diversas formas. En [2] se demostró que la configuración que mejor se adaptó a este sistema se basaba en una malla de clasificadores, repartidos de forma hexagonal.

Para realizar la detección de personas, se utiliza esta malla de clasificadores repartidos a lo largo y ancho de toda la imagen. Cada descriptor se encarga de realizar la detección de una determinada zona de la imagen. Durante el entrenamiento estos clasificadores aprenderán las cualidades que tiene una persona cuando se encuentra en su área de detección.

Para entrenar estos clasificadores se utilizan anotaciones puntuales sobre la cabeza de las personas, que activarán los clasificadores correspondientes a esa posición.

Una vez tenemos todos los clasificadores entrenados, estos calcularán la probabilidad de que la detección sea correcta ante la presencia de una persona. Hay un parámetro llamado *threshold* que indica el umbral mínimo a partir del cual se considera una detección. Al igual que en los métodos de ventanas deslizantes hay que seleccionar los valores máximos para evitar que se produzcan varias detecciones para una sola persona.

El código está programado para que sea necesaria la activación de siete clasificadores de manera simultánea para considerar que se ha producido una detección.

La posición final se calcula mediante la media de las sumas de las probabilidades de los clasificadores implicados en la detección, donde los clasificadores con más peso serán los más cercanos.

$$(x, y) = \left(\sum_{i=0}^{N_n} \alpha_i x_i, \sum_{i=0}^{N_n} \alpha_i y_i\right) \quad (3.1)$$

Siendo N_n el número de clasificadores activos necesarios para considerar que se ha realizado una detección, x_i e y_i las coordenadas del clasificador y α_i la probabilidad que ha recibido el clasificador.

3.1.1. Adaptación Deep Learning

La limitación de la primera versión del código [2] estaba en que no soportaba generalizar ante diversos escenarios, obligando a realizar un entrenamiento individual para cada uno de ellos. Al introducir la adaptación propuesta en [3] este inconveniente queda resulto, permitiendo reducir a un solo entrenamiento común para en todos los escenarios.

3.2. Bases de datos.

Para llevar a cabo este trabajo hemos usado dos bases de datos de secuencias de vídeo tomadas con cámaras omnidireccionales.

3.2.1. BOMNI

Esta base de datos ha sido desarrollada por [28] Bogaziçi Üniversitesi de Turquía. Cuenta con dos posibles bancos de imágenes: uno formado por vídeos tomados desde un plano lateral y otro formado por secuencias desde un plano superior. En este trabajo hemos usado las secuencias del segundo grupo.

Todas las secuencias de esta base de datos vienen acompañadas de un fichero *ground truth* (GT), donde se reflejan las anotaciones cuadro a cuadro de su secuencia correspondiente. En este fichero se refleja el número de cuadro, y además las coordenadas de la esquina superior izquierda y su opuesta de la bounding box cuando existe la presencia de una persona.

Este dataset está formado por un total de veintitrés secuencias, divididas en dos grupos. Pese a que todas ellas se desarrollan en la misma habitación, existen pequeñas diferencias que nos permiten separarlas en dos subgrupos.

Scenario 1: Este sería el primer grupo de secuencias, con un total de cinco. Estas secuencias se caracterizan porque solo se desplaza una persona por el escenario. En cada una de ellas varía tanto el recorrido del actor como el número y la posición de las personas que se encuentran estáticas sentadas en una silla.

Nombre	Nº frames
Top-0	1.001
Top-1	794
Top-2	643
Top-3	907
Top-4	866
Total	4.211

TABLA 3.1: Frames por secuencia scenario 1 de BOMNI.

Scenario 2: En el segundo escenario, sin embargo, contamos con un total de dieciocho secuencias. En este grupo, aparecen tres personas en cada secuencia llevando a cabo diferentes acciones por todo el escenario. Al igual que en el escenario uno, varía el recorrido de los actores, el número y posición de personas sentadas.

Nombre	Nº frames
Top-0	459
Top-1	502
Top-2	497
Top-3	546
Top-4	452
Top-5	533
Top-6	432
Top-7	457
Top-8	557
Top-9	505
Top-10	299
Top-11	416
Top-12	511
Top-13	463
Top-14	485
Top-15	556
Top-16	538
Top-17	499
Total	8.704

TABLA 3.2: Frames por secuencia scenario 2 de BOMNI.

En las secuencias de BOMNI podemos encontrar personas realizando distintas acciones. Por ejemplo, existen secuencias donde el actor se lava las manos, bebe de un

recipiente o conversa con el resto de los actores. Pero la principal característica de las secuencias de esta base de datos es que todas acaban con el actor fingiendo un desmayo en el centro de la habitación. Esto nos supone un pequeño reto a la hora de señalar la posición de la persona, pero lo detallaremos más adelante.

3.2.2. PIROPO

La segunda base de datos empleada en este trabajo es la de PIROPO, cuyo acrónimo significa, *People in Indoor Rooms with Perspective and Omnidirectional cameras*[5].

Esta base de datos cuenta con secuencias de vídeo tomadas con cámaras omnidireccionales en dos habitaciones distintas. Podemos encontrar a los actores realizando tres tipos de acciones: manteniéndose estáticos durante toda la secuencia, desplazándose a lo largo de toda la habitación o sentados.

Al igual que en la base de datos de BOMNI, todas las secuencias tienen asociadas un fichero con las anotaciones GT correspondientes a cada cuadro. En esta ocasión las anotaciones proporcionadas vienen en formato puntual.

Hemos utilizado un total de diecinueve secuencias de la base de datos de PIROPO, de las cuales cuatro secuencias han sido destinadas al proceso de entrenamiento y las quince restantes han sido destinadas para test.

Secuencias de entrenamiento.

Como ya se ha comentado, existen cuatro secuencias de entrenamiento. En la Tabla 3.3 analizamos cada una de las secuencias de este grupo.

Estas secuencias se caracterizan porque siempre hay una persona desplazándose por la habitación, repitiéndose este proceso con tres actores distintos.

Nombre	Nº frames
Cámara 1A	10.969
Cámara 1B	4.584
Cámara 2A	10.401
Cámara 1A	10.975
Total	36.929

TABLA 3.3: Frames por cada secuencia de entrenamiento de PIROPO.

Secuencias de test

En estas secuencias es donde encontramos mayor variabilidad, lo que supone una gran ventaja para evaluar el rendimiento de nuestro sistema, dado que nos permite ver si es capaz de generalizar ante distintos casos y condiciones.

Contamos con cuatro secuencias de test para cada secuencia de entrenamiento, exceptuando el caso de la secuencia 1B que solo tenemos tres.

Nombre	Características
Test-1	Una persona presente durante la etapa de entrenamiento caminando por la habitación.
Test-2	Tres personas caminando simultáneamente por la habitación.
Test-3	Tres personas caminado por la habitación, estando dos de ellas presentes durante el proceso de entrenamiento.
Test-4	Una persona caminando por la habitación y permaneciendo estática durante un periodo de tiempo.

TABLA 3.4: Características secuencias de test PIROPO.

Como hemos hecho anteriormente en la Figura, 3.5 hemos reflejado la cantidad de cuadros que tenemos para evaluar la calidad del entrenamiento en el próximo capítulo.

Nombre	Nº frames
Test 1 1A	2.270
Test 2 1A	1.325
Test 3 1A	1.782
Test 4 1A	895
Test 1 1B	518
Test 2 1B	267
Test 3 1B	378
Test 1 2A	2.201
Test 2 2A	1.401
Test 3 2A	1.646
Test 4 2A	831
Test 1 3A	2.260
Test 2 3A	1.389
Test 3 3A	1.579
Test 4 3A	301
Total	29.066

TABLA 3.5: Frames por secuencia test de PIROPO.

3.2.3. Adaptaciones para nuestro sistema.

Para poder trabajar con ambas bases de datos de manera conjunta hemos tenido que introducir pequeños cambios en el formato de nuestras bases de datos. Bien es cierto que estos pequeños cambios solo fueron aplicados a BOMNI, dado que PIROPO ya había sido adaptado anteriormente en [3].

Adaptaciones en secuencias de vídeo

El sistema que hemos empleado para realizar este trabajo recibe como parámetros de entrada imágenes en formato JPEG, con un ratio 1:1. En BOMNI nuestro punto de partida eran secuencias de vídeo en formato MP4. Por tanto, lo primero que tuvimos que hacer fue extraer cada uno de los cuadros de cada secuencia de manera totalmente independiente, para ello se utilizó la herramienta de software libre ffmpeg.

Los cuadros originales tenían una resolución de 640x480 píxeles. Con esta resolución todavía no tenían el formato deseado, por lo que, mediante un scrypt de MATLAB, recortamos las imágenes para lograr la resolución de 480x480.

Imagen A Imagen B

FIGURA 3.2: Imagen A, frame original extraído de BOMNI. Imagen B, frame adaptado al sistema.

Adaptaciones en anotaciones

Las anotaciones aportadas por el dataset de BOMNI mantenían el formato Bounding Box. No obstante el formato empleado por nuestro sistema se corresponde con formato puntual, el cual resalta la presencia de una persona remarcando su cabeza mediante un punto. El formato original de BOMNI sigue la siguiente estructura: un identificador para cada frame, las coordenadas del vértice superior izquierda (xi,yi) y las del vértice opuesto (xd,yd).

Para poder emplear el dataset de BOMNI en nuestro sistema tuvimos que transformar las anotaciones dadas en anotaciones puntuales.

Como vimos en la Sección 2.2.1 del capítulo 2 de este trabajo las imágenes omnidireccionales son de carácter esférico e introducen mayor distorsión en los bordes de la imagen. Esta distorsión se traduce en que la cabeza de una persona se alejara más del centro del bounding box cuanto más al borde de la imagen se encuentre. Conociendo estas limitaciones realizamos la siguiente suposición: la cabeza de la persona siempre recae sobre una linea imaginaria que une el centro de su bounding box y el centro de la imagen.

La distancia entre el centro de la imagen y el centro de la bounding box determina el factor de corrección que hay que aplicar para conocer la posición final de la cabeza.

Esta aproximación ofreció buenos resultados en la mayoría de situaciones, excepto cuando la persona variaba su postura corporal, en la Figura 3.3 se puede ver un ejemplo de los resultados obtenidos.

FIGURA 3.3: Ejemplo de las dificultades para calcular la posición de la cabeza cuando la persona adopta posiciones corporales fuera de lo común.

Procedimiento

El primer paso para llevar a cabo esta transformación seria dividir la imagen como si de un sistema de coordenadas se tratase, haciendo coincidir el centro geométrico con el origen de coordenadas. Esto nos permite calcular una linea imaginaria que una el centro de la imagen y la posición final de la cabeza, asumiendo que esta pasa por el centro de la bounding box.

Mediante las coordenadas proporcionadas por BOMNI, calculamos el centro de la bounding box, para posteriormente determinar la distancia existente entre este punto y el centro de la imagen.

Por otra parte, para calcular el factor de corrección se tenían en cuanta la distancia de la persona respecto al origen, el tamaño de la bounding box y el radio total de la esfera.

Finalmente, la posición de la cabeza se determina al calcular la distancia mediante la suma de las dos distancia anteriores. Obteniendo así las anotaciones en formato puntual.

La finalidad del trabajo, desde el primer momento ha sido dotar al sistema de la capacidad de detectar todas las personas presentes en la imagen. En las anotaciones de BOMNI no estaban incluidas las personas que permanecían estáticas en las secuencias, por lo que una vez ya tuvimos las anotaciones en formato puntual añadimos las anotaciones correspondientes a esas personas.

3.3. Técnica de aumento de datos

Tras el primer entrenamiento realizado con BOMNI, observamos que la red no era capaz de generalizar ante nuevas secuencias, en la Sección 4.3 del capítulo cuatro se pueden encontrar más detalles. Para solucionar este inconveniente aplicamos técnicas de aumento de datos, con el objetivo de aumentar la variabilidad y la cantidad de imágenes para las secuencias de entrenamiento. La idea principal consiste en generar nuevas imágenes de entrenamiento añadiendo personas extraídas de otros frames. En la Figura 3.4 podemos ver un ejemplo de los resultados obtenidos tras aplicar estas técnicas.

FIGURA 3.4: La imagen de la izquierda se corresponde con una imagen original de una secuencia de BOMNI. La imagen de la derecha ha sido creada con técnicas de aumento de datos a partir de cuatro imágenes de la secuencia original.

Para llevar a cabo este proceso, se asume que los 10 primeros frames de las secuencias, están libres de la presencia de personas y por lo tanto estas imágenes son usadas como fondo para las nuevas. De forma aleatoria en el resto de imágenes de las secuencia se aplica un modelo gaussiano de background substraction [29], que nos permitía extraer la silueta de una persona y combinarla con las imágenes base. En la Figura 3.5, se reflejan los pasos a seguir al aplicar este algoritmo.

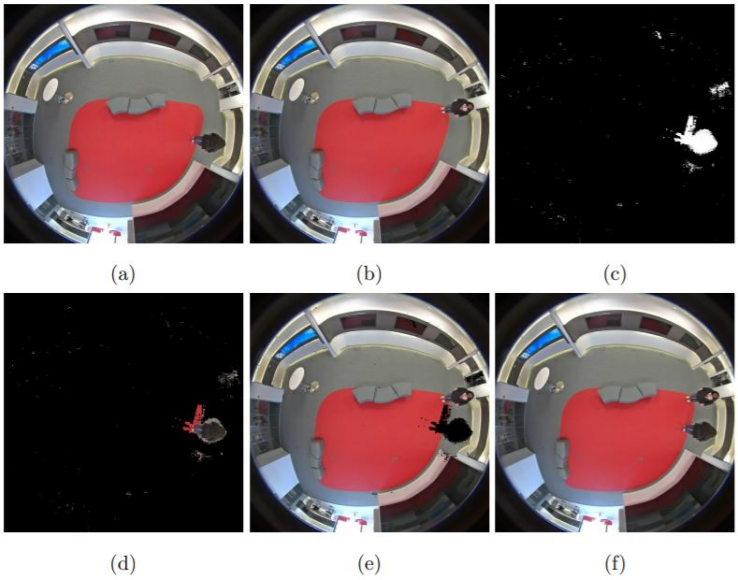

FIGURA 3.5: Extraída de [3]. Las imágenes (a), (b) corresponden con frames originales de PIROPO, tras la aplicación del algoritmo background substraction obtenemos un filtro de la persona deseada (c), (d) con el filtro obtenido extraemos a la persona de la imagen original y posteriormente aplicamos el filtro inverso sobre la imagen de destino (d) para evitar problemas a la hora de añadir la persona extraida. En (f) podemos observar la imagen generada tras los pasos anteriores.

Durante la elaboración del trabajo aplicamos esta técnica a todas las secuencias de entrenamiento de BOMNI variando el número de frames de los cuales se realizará la extracción de personas y la longitud final de las nuevas secuencias, en la Sección 4.4 próximo captulo se ofrece un estudio detallado para cada uno de los casos realizados.

3.4. Aumento del número de clasificadores

El Grid de 825 clasificadores propuesto inicialmente, no ofrecía la suficiente resolución para diferenciar la detección de dos personas próximas espacialmente. El área de detección para cada clasificador es muy grande para diferenciar en estas situaciones. Cada clasificador solo puede asociarse a una detección, en el caso de que un clasificador se active por la presencia de varias personas este solo va a poder detectar una, eliminando durante la etapa de NMS las de menor valor.

En la esquina superior izquierdad de la Figura 3.6 podemos apreciar un ejemplo claro del problema de resolución del Grid original. Con esta resolución los clasificadores solapan la detección de las dos personas.

FIGURA 3.6: Problema de la resolución en el Grid de clasificadores al detectar personas próximas espacialmente.

Para solucionar este problema, se propone aumentar el número de clasificadores hasta 3300, lo que conlleva un aumento en un factor 4 respecto al Grid inicial. En la Figura 3.7, se realiza una comparación entre los dos modelos propuestos.

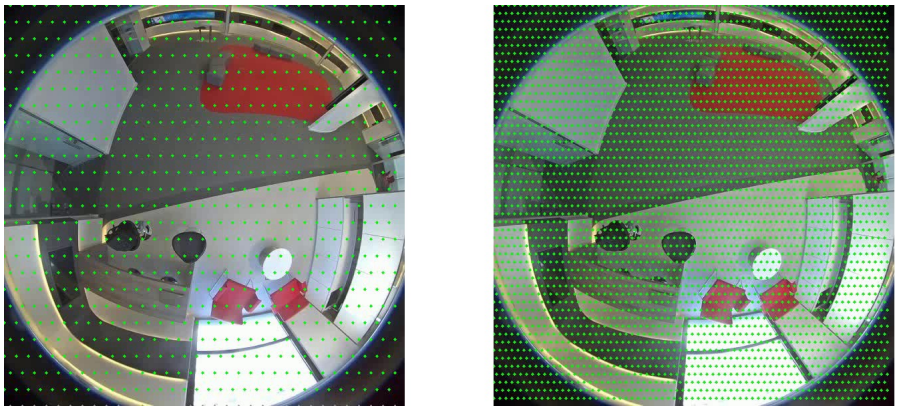

FIGURA 3.7: La imagen de la izquierda se corresponde con el grid de 825 clasificadores inicial, la imagen de la derecha es el grid de 3300 clasificadores propuesto.

El aumento del número de clasificadores aporta un aumento en la precisión de detección de nuestro sistema. Al contar con un mayor número de clasificadores el área de detección para cada uno de ellos se ve reducida, esto conlleva que los clasificadores aprendan características muy específicas, pudiendo ser un problema a la hora de generalizar a nuevos escenarios. Es importante encontrar un equilibrio entre la precisión de nuestro sistema y el número de clasificadores.

Al realizar esta modificación, tuvimos que adaptar parte de nuestro sistema de detección. Al tener más clasificadores, la última capa de nuestra red neuronal, la capa fully

conected aumenta también el número de neuronas de salida hasta 3300. En los dataset empleados para el trabajo no tuvimos que añadir modificaciones, pero si generar de nuevo las activaciones de las secuencias de entrenamiento para llevarlo a cabo.

En la Sección 4.5.2 del Capítulo 4 de este trabajo, se reflejan los resultados de detección de este sistema con el Grid de 3300 clasificadores y se puede comprobar el aumento de la precisión debido a esta modificación.

Capítulo 4

Experimentos y resultados

En este capítulo del trabajo, vamos a exponer y razonar los resultados obtenidos durante todo el proceso de investigación.

4.1. Métricas de rendimiento

Este detector se enfrenta a un problema de localización y clasificación binaria. Es decir, tenemos que diferenciar entre dos clases que vamos a definir de la siguiente manera.

- **Clase negativa:** No se detecta presencia de personas, por lo que el clasificador permanece inactivo.
- **Clase positiva:** Se detecta la presencia de personas, por tanto, el clasificador se activa.

Teniendo esta diferenciación entre las distintas clases claras, vemos que durante la clasificación se pueden dar cuatro posibles combinaciones. Que dan lugar a los siguientes parámetros.

- **Verdadero positivo (TP):** El clasificador etiqueta como positivo un elemento de la clase positiva.
- **Falso positivo (FP):** El clasificador etiqueta como positivo un elemento de la clase negativa.
- **Verdadero negativo (TN):** El clasificador etiqueta como negativo un elemento de la clase negativa.
- **Falso negativo (FN):** El clasificador etiqueta como negativo un elemento de la clase positiva.

En el caso de este trabajo, se considerara un como un verdadero positivo si la distancia entre la detección y el punto real calculado a partir de GT es mayor que un umbral definido como d. Para el desarrollo de este trabajo hemos fijado la distancia d en 45 píxeles.

Utilizando estos parámetros [30] de base, existen unas fórmulas que nos permiten evaluar el rendimiento del sistema total:

- **Precisión:** Este parámetro permite medir la precisión de nuestro sistema, es decir, que porcentaje de acierto hay ante todos los datos que tenemos detectados como positivos y los que realmente lo son.

$$\text{Precisión} = \frac{TP}{TP+FP}$$

- **Recall:** Este parámetro permite medir la exhaustividad de nuestro sistema, es decir, que porcentaje de acierto hay ante todos los datos que tenemos detectado como positivos.

$$\text{Recall} = \frac{TP}{TP+FN}$$

- **Fscore:** Este parámetro es una combinación de los dos anteriores, permitiendo evaluar de forma única los dos parámetros.

$$\text{Fscore} = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

4.2. Separación dataset

En la sección anterior, vimos que la base de datos de PIROPO contaba con una separación previa entre secuencias de entrenamiento y secuencias de Test. Durante el desarrollo de este trabajo decidimos mantener esta separación para que los resultados obtenidos fuesen una continuación de los trabajos anteriores y así medir como afectan los cambios introducidos.

En el caso de BOMNI, no había esa separación previa, además, la diferencia entre las secuencias dentro de cada escenario era el actor que realizaba cada acción, pero todas siguen un patrón casi idéntico.

Teniendo en cuenta la longitud de las secuencias de ambos escenarios de esta segunda base de datos, se decidió destinar diecisiete secuencias a la etapa de entrenamiento y las seis restantes al proceso de test, quedando la distribución final como se refleja en la Tabla 4.1.

	Train	Test
Scenario 1	Top-0	Top-4
	Top-1	
	Top-2	
	Top-3	
Scenario 2	Top-0	Top-4
	Top-1	Top-5
	Top-2	Top-6
	Top-3	Top-7
	Top- 9	Top-8
	Top- 10	
	Top- 11	
	Top- 12	
	Top- 13	
	Top- 14	
	Top- 15	
	Top- 16	
	Top- 17	

TABLA 4.1: División de secuencias entrenamiento y test para la base de datos BOMNI.

4.3. Primeras Aproximaciones

En esta primera fase del trabajo entrenamos tres modelos independientes con las secuencias originales de cada dataset.

Para el primer modelo, se utilizaron solamente las secuencias destinadas a la fase de entrenamiento de PIROPO, obteniendo los resultados reflejados de la Tabla 4.2 para las secuencias de Test.

El segundo modelo, fue entrenado con las secuencias de entrenamiento descritas en el anterior apartado de la base de datos de BOMNI. En la Tabla 4.3 vemos que lo resultados obtenidos no son tan buenos como en el caso del modelo anterior.

	Secuencia	Precisión (%)	Recall (%)	Fscore(%)
Cámara 1A	Test 1	88,92	88,87	88,99
	Test 2	82,68	27,33	41,08
	Test 3	93,63	73,97	82,65
	Test 4	77,67	42,36	54,82
Cámara 2A	Test 1	93	92,93	92,96
	Test 2	80,26	23,71	36,6
	Test 3	61,62	89,72	73,06
	Test 4	69,33	89,8	78,25
Cámara 3A	Test 1	94,74	94,97	94,85
	Test 2	92,14	53,75	67,89
	Test 3	75	0,89	1,76
	Test 4	88,37	17,12	28,68
Cámara 1B	Test 1	96,95	94,85	95,89
	Test 2	87,77	27,42	41,78
	Test 3	96,49	85,05	85,76
Rendimiento global		84,84	61,97	65,87

TABLA 4.2: Rendimiento del sistema de detección utilizando solamente el dataset de entrenamiento de PIROPO.

	Secuencia	Precisión (%)	Recall (%)	Fscore(%)
Scenario 1	Top-4	75,89	69,75	72,69
Scenario 2	Top-4	11,47	16,03	13,37
	Top-5	19,44	22,7	20,94
	Top-6	33,79	25,43	29,02
	Top-7	35,73	24,68	29,19
	Top-8	38	27,36	31,81
Rendimiento global		35,72	30,99	32,83

TABLA 4.3: Rendimiento del sistema de detección utilizando solamente el dataset de entrenamiento de BOMNI.

Como la finalidad de este trabajo era medir el rendimiento total del sistema cuando se utilizaban ambas bases de datos de manera conjunta, el tercer modelo neuronal se entrenó con las secuencias de entrenamiento de PIROPO y BOMNI, obteniendo los resultados de la Tabla 4.4

	Secuencia	Precisión (%)	Recall (%)	Fscore(%)
PIROPO				
Cámara 1A	Test 1	86,08	88,02	87,04
	Test 2	78,04	24,48	37,27
	Test 3	64,41	73,69	84,747
	Test 4	75,59	66,58	70,8
Cámara 2A	Test 1	91,3	91,63	91,47
	Test 2	80,57	24,17	37,19
	Test 3	61,34	83	70,54
	Test 4	65,76	81,28	72,7
Cámara 3A	Test 1	92,58	93,65	93,11
	Test 2	88,71	54,01	67,14
	Test 3	83,33	1,48	2,92
	Test 4	87,5	3,15	6,09
Cámara 1B	Test 1	94,65	95,93	95,29
	Test 2	78,83	24,27	37,11
	Test 3	78,41	78,41	78,41
Rendimiento PIROPO		80,47	58,91	62,05
BOMNI				
Scenario 1	Top-4			
Scenario 2	Top-4	23,22	26,79	23,98
	Top-5	46,43	22,49	30,3
	Top-6	42,73	20,27	27,49
	Top-7	44,84	18,93	26,63
	Top-8	53	20,94	30,02
Rendimiento BOMNI		42,04	21,48	27,68
Rendimiento global		70,86	33,43	40,42

TABLA 4.4: Rendimiento del sistema de detección utilizando los dataset de BOMNI y PIROPO de manera simultánea.

Estas primeras aproximaciones permitieron darnos cuenta de que para el caso de PIROPO los resultados eran generalmente buenos y de que el sistema obtenido era capaz de generalizar ante nuevas secuencias.

Sin embargo, para el caso de BOMNI, como podemos observar en la Figura 4.1 , el modelo neuronal se entrenaba bien, pero este no era capaz de generalizar ante los nuevos escenarios.

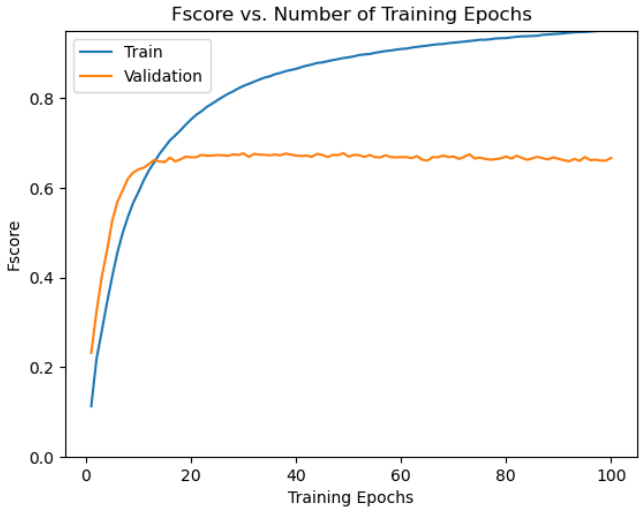

FIGURA 4.1: Fscore obtenido durante el entrenamiento de la red neuronal con las secuencias originales.

4.4. Mejora de rendimiento en BOMNI

Tras los resultados obtenidos durante la primera fase del trabajo, observamos que no eran válidos en el caso de BOMNI, por lo que debíamos mejorar este modelo de manera independiente para obtener unos resultados concluyentes en el trabajo final.

Para ello, en esta segunda fase de la investigación se aplicó la técnica de aumento de datos en las secuencias de entrenamiento comentada en la Sección 3.3, con la finalidad de aumentar la variabilidad en las secuencias de entrenamiento, intentando dotar al sistema de detección de la capacidad de generalizar ante nuevas situaciones.

Este tipo de métodos normalmente es realizado por la propia CNN al recibir las secuencias de entrenamiento, sin embargo en este trabajo se ha realizado de manera local dando lugar a una mayor diversidad en el dataset de entrenamiento.

Como había muchas combinaciones posibles, primero nos centramos en encontrar el número de mezcla de imágenes que ofrecía mejor rendimiento para obtener las nuevas secuencias.

Para crear estos escenarios, en la longitud inicial se mantuvo un constante aumento de cuatro veces la longitud de las secuencias originales, variando en dos, cuatro y seis el número de frames, utilizados para crear las nuevas imágenes.

Tras los entrenamientos correspondientes se obtuvieron los resultados representados en la Tabla 4.5

		Frames x4 , 2 Frames de mezcla			Frames x4 , 4 Frames de mezcla			Frames x4 , 6 Frames de mezcla		
	Nombre	Precisión(%)	Recall(%)	Fscore(%)	Precisión(%)	Recall(%)	Fscore(%)	Precisión(%)	Recall(%)	Fscore(%)
Scenario 1	Top-4	97,06	91,18	94,03	97,03	90,48	93,64	98,32	92,94	95,55
Scenario 2	Top-4	69,36	60,85	64,83	69,47	61,35	65,16	69,21	60,6	64,62
	Top-5	69,26	61,03	64,88	69,01	60,34	65,16	70,52	63,97	67,08
	Top-6	62,16	58,62	60,34	60,28	58,42	64,38	61,67	57,72	59,63
	Top-7	62,23	55,41	58,62	62,23	55,41	59,34	62,63	55,66	58,94
	Top-8	65,07	59,47	62,43	65,82	59,23	62,35	66,07	59,32	62,52
Media global		70,96	64,43	67,52	70,64	64,20	67,24	**71,40**	**65,03**	**68,06**

TABLA 4.5: Comparación de resultados obtenidos variando el número de frames implicados en la creación de las nuevas imágenes en las secuencias de entrenamiento de BOMNI.

Analizando los resultados de cada entrenamiento observamos que los mejores resultados para este lote se obtienen cuando empleamos seis frames de mezcla para crear las nuevas secuencias. Por esta razón, durante esta segunda etapa decidimos mantener los seis frames de mezcla en los nuevos casos, variando solamente la longitud de las secuencias respecto a las originales.

Una vez fijados seis frames utilizados en la mezcla probamos a entrenar la red neuronal con estas nuevas condiciones, aumentando en un factor, dos, cuatro y seis la longitud de las nuevas secuencias respecto a las originales. En la Tabla 4.6 vemos los resultados obtenidos.

		Frames x2 , 6 Frames de mezcla			Frames x4 , 6 Frames de mezcla			Frames x6 , 6 Frames de mezcla		
	Nombre	Precisión(%)	Recall(%)	Fscore(%)	Precisión(%)	Recall(%)	Fscore(%)	Precisión(%)	Recall(%)	Fscore(%)
Scenario 1	Top-4	97,13	92,83	94,93	98,32	92,94	95,55	97,03	90,50	93,66
Scenario 2	Top-4	69,13	60,02	64,25	69,21	60,60	64,62	69,40	61,52	65,22
	Top-5	70,13	62,15	65,90	70,52	63,97	67,08	69,57	61,31	65,18
	Top-6	62,07	57,98	59,95	61,67	57,72	59,63	61,72	58,55	60,09
	Top-7	62,38	55,47	58,73	62,63	55,66	58,94	63,04	56,75	59,73
	Top-8	66,16	59,56	62,69	66,07	59,32	62,52	66,17	59,76	62,80
Media global		71,16	64,66	67,74	**71,40**	**65,03**	**68,06**	71,15	64,73	67,78

TABLA 4.6: Comparación de resultados variando las de la longitud de las nuevas secuencias de entrenamiento de BOMNI.

Observando gráfica y estadísticamente los resultados finales de este conjunto de entrenamientos, determinamos que los mejores resultados se obtienen cuando empleamos la siguiente configuración: aumentando en un factor cuatro la longitud de la nueva secuencia respecto a la original y utilizando seis frames de ésta para crear cada una de las nuevas imágenes.

En esta fase de la investigación observando los resultados gráficos obtenidos identificamos que la principal fuente de errores viene determinada por la resolución del Grid de clasificadores empleado.

En BOMNI disponemos de dos escenarios. En el primero, solamente una persona se desplaza por la habitación y esto no supone ningún problema ni en la etapa de detección ni en la de clasificación. En el segundo escenario, son varias personas las que se mueven

a través de la habitación donde el sistema propuesto es capaz de detectarlas a todas sin dificultad alguna, pero en la etapa de clasificación vimos que el Grid de clasificadores propuesto no tenía la suficiente resolución como para separar la detección de dos personas próximas. En la Figura 3.6, podemos ver un ejemplo claro de este problema en la esquina superior izquierda de la imagen.

4.5. Resultados finales

Al evaluar los resultados en la Sección anterior se nos abrió una segunda vía de investigación. La primera nos permitía continuar con el objetivo original del trabajo, evaluar el rendimiento total del sistema cuando empleábamos los dataset de BOMNI y PIROPO de manera conjunta.

La segunda consistía en comprobar qué pasaba al aumentar la resolución del Grid de clasificadores empleado.

4.5.1. Resultados PIROPO y BOMNI

Una vez se logró obtener resultados satisfactorios para cada dataset de forma independiente, se entrenó un modelo neuronal que emplease las secuencias de entrenamiento de ambas bases de datos de manera conjunta. En la Tabla 4.7 se reflejan los resultados obtenidos.

Para analizar el rendimiento del sistema conjunto, hemos decidido usar como referencia el valor de Fscore, como hemos comentado anteriormente, este valor unifica los resultados de Recall y precisión del sistema.

Para las secuencias de test de la base de datos de PIROPO vemos que el promedio de Fscore aumenta en un 0,8 % su efectividad respecto al valor obtenido inicialmente.

Esta diferencia se hace mucho más evidente en las secuencias de test de BOMNI, donde el valor de Fscore aumenta en un 39,95 % respecto al valor obtenido en el entrenamiento con las secuencias originales.

El rendimiento de manera conjunta mejora el valor de Fscore en un 23,8 %. Analizando de forma paralela los resultados gráficos que se obtienen, se puede apreciar el buen funcionamiento del sistema para las secuencias de test.

Analizando los resultados, podemos afirmar que dotando al sistema de más secuencias de entrenamiento y con más variabilidad en ellas, aumentan los resultados en la detección final.

	Secuencia	Precisión (%)	Recall (%)	Fscore(%)
PIROPO				
Cámara 1A	Test 1	98,92	98,02	98,47
	Test 2	98,9	26,26	41,5
	Test 3	96,54	23,76	38,14
	Test 4	96,3	10,1	18,29
Cámara 2A	Test 1	99,71	98,7	99,2
	Test 2	98,87	30,51	46,63
	Test 3	93,97	90,46	92,18
	Test 4	92,79	94,06	93,42
Cámara 3A	Test 1	98,27	98,1	98,18
	Test 2	98,92	57,29	72,56
	Test 3	100	1,34	2,64
	Test 4	100	1,8	3,54
Cámara 1B	Test 1	96,83	99,46	98,13
	Test 2	96,61	25,62	40,5
	Test 3	99,01	100	99,5
Rendimiento PIROPO		97,70	57,03	62,85
BOMNI				
Scenario 1	Top-4	98,17	92,92	95,47
Scenario 2	Top-4	68,87	59,27	63,71
	Top-5	69,61	61,73	65,43
	Top-6	61,75	57,4	59,5
	Top-7	62,83	56,44	59,46
	Top-8	65,91	58,89	62,2
Rendimiento BOMNI		71,19	64,44	67,63
Rendimiento global		90,13	59,15	64,22

TABLA 4.7: Resultados obtenidos empleando PIROPO y las secuencias ampliadas de BOMNI.

Es importante recalcar que todavía no se cuenta con la resolución suficiente para poder detectar dos personas espacialmente próximas entre sí, en las secuencias de BOMNI.

4.5.2. Aumento de resolución del Grid de clasificadores.

El objetivo de introducir este cambio es dotar al sistema de la capacidad de diferenciar la detección de dos personas próximas entre sí, logrando así una detección más precisa para las secuencias de BOMNI.

Para medir el rendimiento del sistema anterior, con este nuevo de Grid clasificadores, se realizó un primer entrenamiento utilizando solamente las secuencias de entrenamiento de BOMNI, anteriormente descritas. En la Tabla 4.8 podemos observar los resultados obtenidos con este modelo.

	Secuencia	Precisión (%)	Recall (%)	Fscore(%)
Scenario 1	Top-4	98,35	93,65	95,94
Scenario 2	Top-4	70,28	67,7	68,96
	Top-5	69,65	69,06	69,35
	Top-6	64,01	64,38	64,2
	Top-7	65,45	68,82	64,62
	Top-8	68,15	65,91	67,01
Rendimiento global		63,95	64,55	71,68

TABLA 4.8: Rendimiento del sistema de detección utilizando solamente el dataset de entrenamiento de BOMNI y un grid con 3300 clasificadores.

Como se puede apreciar, con el nuevo Grid de clasificadores existe una leve mejoría en las estadísticas de detección para las secuencias de BOMNI. El Fscore obtenido con este modelo neuronal aumenta un 3,62 %. En la Figura vemos gráficamente como se soluciona el problema de la resolución expuesto anteriormente.

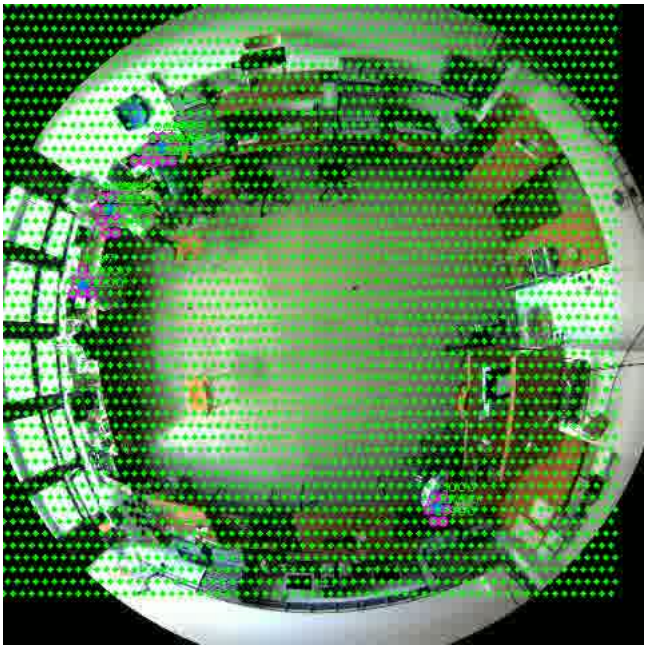

FIGURA 4.2: Utilizando un Grid con mayor resolución, se puede apreciar que los problemas generados por las detecciones espacialmente próximas quedan resueltos.

Para concluir con la investigación decidimos ampliar el estudio realizando un último entrenamiento que implicase ambas bases de datos utilizando el nuevo Grid planteado en la etapa de detección. En la Tabla 4.9 se reflejan los resultados obtenidos para este caso.

	Secuencia	Precisión (%)	Recall (%)	Fscore(%)
PIROPO				
Cámara 1A	Test 1	98,55	95,71	97,11
	Test 2	97,91	16,66	28,47
	Test 3	92	1,63	3,21
	Test 4	100	1,3	2,56
Cámara 2A	Test 1	99,96	96,32	97,95
	Test 2	99,37	18,5	31,19
	Test 3	79,11	47,1	59,04
	Test 4	84,96	29,22	43,49
Cámara 3A	Test 1	98,32	91,92	95,01
	Test 2	99,25	41,35	58,28
	Test 3	0	0	0
	Test 4	0	0	0
Cámara 1B	Test 1	98,06	95,93	96,99
	Test 2	0	0	0
	Test 3	100	92,36	96,03
Rendimiento PIROPO		71,72	39,25	44,34
BOMNI				
Scenario 1	Top-4	97,8	94,91	96,34
Scenario 2	Top-4	69,48	65,19	67,27
	Top-5	69,24	67,74	68,48
	Top-6	63,34	62,2	62,77
	Top-7	65,02	62,73	63,85
	Top-8	67,79	64,21	65,95
Rendimiento BOMNI		72,11	69,50	70,78
Rendimiento global		75,25	49,76	54

TABLA 4.9: Resultados obtenidos empleando PIROPO y las secuencias ampliadas de BOMNI utilizando un Grid de 3300 clasificadores.

Cuando analizamos el sistema empleando ambos dataset, observamos que el rendimiento individual de BOMNI aumenta un 3,25 % su fscore. Sin embargo, el rendimiento de PIROPO decae un 18,51 % y el rendimiento global como consecuencia también se reduce en un 10,22 %.

Esta bajada tan brusca en el rendimiento del sistema es posible que este relacionada con un sobre ajuste a los datos de entrenamiento.

Capítulo 5

Conclusiones y trabajos futuros

5.1. Conclusiones

Al comienzo de este trabajo se sabía que el sistema desarrollado en [3] obtenía buenos resultados al ser entrenado con el dataset de PIROPO. Nuestro objetivo durante el desarrollo fue mejorar los resultados de trabajos anteriores y evaluar que efecto generaban los cambios introducidos.

Una de las medidas adoptadas fue, ampliar el número de escenarios durante el entrenamiento, para ello añadimos el dataset de BOMNI. Tras analizar los resultados obtenidos a partir de un modelo neuronal entrenado con las secuencias originales de este dataset vimos que no se obtenían buenos resultados durante la detección. Para solucionar estos problemas aplicamos técnicas de aumento de datos en las secuencias de entrenamiento, buscando cual era la combinación que ofrecía mejores resultados. Otro de los cambios que evaluamos consistía en cuadriplicar el número de clasificadores implicados en la etapa de la detección con el fin de aumentar la precisión en la localización de las personas sobre la imagen.

Una vez que conseguimos buenos resultados en la detección de manera individual para cada dataset evaluamos que impacto tenia ampliar el número de escenarios y entrenar un modelo neuronal con ambos dataset de manera conjunta.

Tras estudiar cada uno de los caso anteriormente propuestos, llegamos a las siguientes conclusiones:

Al aumentar el número de imágenes de entrenamiento, como es lógico, aumenta también el coste computacional de este proceso. Es en este punto donde debemos plantearnos si los beneficios que obtenemos son lo suficientemente relevantes como para asumir estos costes.

Al aumentar el Grid de clasificadores, conseguimos mayor precisión al localizar personas espacialmente próximas. Como en el caso anterior esto supone un aumento en el

coste computacional del entrenamiento.

Cuantas más secuencias de entrenamiento aportemos a nuestro detector, mejores resultados se obtendrán cuando el sistema se enfrente a nuevos escenarios.

Si comparamos los distintos resultados obtenidos durante el entrenamiento observamos que sino aplicamos técnicas de aumento de datos a las secuencias de BOMNI no habría sido posible que el sistema generalizase ante las secuencias de test.

Analizando las estadísticas de los distintos modelos neuronales se puede ver que entrenar con ambos dataset de manera conjunta no implica un aumento en el rendimiento frente a los modelos individuales, pero es cierto que este sistema resultante es capaz de generalizar ante escenarios de ambos dataset. Esto permite tener un solo modelo que obtenga buenos resultados ante distintos escenarios, en el sistema propuesto en [2] esto no era posible, ya que había que tener un clasificador entrenado para cada escenario.

5.2. Trabajos futuros

Una posible vía de investigación sería investigar cual es el número de clasificadores óptimos para que el Grid de clasificación pueda realizar las detecciones de manera precisa.

También se propone analizar que resultados se obtienen al cambiar el modelo de CNN utilizando por ejemplo Vgg, Resnet50 o cualquier modelo destinado al tratamiento de imágenes.

Unificar el sistema de detección en un único entorno de programación es una opción muy interesante, actualmente se utilizan C++ y Phyton.

Para terminar, también se propone ampliar el número de dataset para añadir mayor variabilidad en la etapa de entrenamiento, otros posibles dataset serian: SLAM [31] o OmniDepth [32].

Bibliografía

[1] Fatih Porikli, Francois Bremond, Shiloh L Dockstader, James Ferryman, Anthony Hoogs, Brian C Lovell, Sharath Pankanti, Bernhard Rinner, Peter Tu y Péter L Venetianer. «Video surveillance: past, present, and now the future [DSP Forum]». En: *IEEE Signal Processing Magazine* 30.3 (2013), págs. 190-198.

[2] Carlos R del-Blanco, Pablo Carballeira, Fernando Jaureguizar y Narciso Garcıa. «Robust people indoor localization with omnidirectional cameras using a Grid of Spatial-Aware Classifiers». En: *Signal Processing: Image Communication* 93 (2021), pág. 116135.

[3] Enrique SEPÍLVEDA JORCANO y col. «People detection in omnidirectional cameras: development of a deep learning architecture based on a spatial grid of classifiers». B.S. thesis. 2020.

[4] Eduardo Rivera y col. «Introducción a las Redes Neuronales Artificiales.» En: (2015).

[5] Carlos R. del-Blanco, Pablo Carballeira, Fernando Jaureguizar y Narciso García. «Robust people indoor localization with omnidirectional cameras using a Grid of Spatial-Aware Classifiers». En: *Signal Processing: Image Communication* 93 (2021), pág. 116135. ISSN: 0923-5965. DOI: https://doi.org/10.1016/j.image.2021.116135. URL: https://www.sciencedirect.com/science/article/pii/S0923596521000023.

[6] Mykhaylo Andriluka, Stefan Roth y Bernt Schiele. «People-tracking-by-detection and people-detection-by-tracking». En: *2008 IEEE Conference on computer vision and pattern recognition*. IEEE. 2008, págs. 1-8.

[7] Rodolfo Martın Villanueva. «Detección de objetos por computador». En: (2017).

[8] Pengyu Zhao, Ansheng You, Yuanxing Zhang, Jiaying Liu, Kaigui Bian y Yunhai Tong. «Reprojection R-CNN: A Fast and Accurate Object Detector for 360° Images». En: *arXiv e-prints* (2019), arXiv-1907.

[9] Itsaso Usunariz Arcauz. «Restauración de imágenes mediante pirámide Gaussiana y técnicas de reducción y ampliación». En: (2014).

[10] Navneet Dalal y Bill Triggs. «Histograms of oriented gradients for human detection». En: *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*. Vol. 1. Ieee. 2005, págs. 886-893.

[11] Constantine Papageorgiou y Tomaso Poggio. «A trainable system for object detection». En: *International journal of computer vision* 38.1 (2000), págs. 15-33.

[12] Gustavo A Betancourt. «Las máquinas de soporte vectorial (SVMs)». En: *Scientia et technica* 1.27 (2005).

[13] K-R Müller, Alexander J Smola, Gunnar Rätsch, Bernhard Schölkopf, Jens Kohlmorgen y Vladimir Vapnik. «Predicting time series with support vector machines». En: *International Conference on Artificial Neural Networks*. Springer. 1997, págs. 999-1004.

[14] Leon O Chua. «CNN: A vision of complexity». En: *International Journal of Bifurcation and Chaos* 7.10 (1997), págs. 2219-2425.

[15] Andres Nacelle y E Mizraji. «Redes neuronales artificiales». En: *de Las redes neuronales de la biologıa en algoritmos de clasificación, Uruguay* (2009).

[16] Fernando Giménez Palomares, Juan Antonio Monsoriu Serrá y Elena Alemany Martınez. «Aplicación de la convolución de matrices al filtrado de imágenes». En: *Modelling in Science Education and Learning* 9.1 (2016), págs. 97-108.

[17] Benjamin Graham. «Fractional max-pooling». En: *arXiv preprint arXiv:1412.6071* (2014).

[18] Deqing Sun, Jonas Wulff, Erik B Sudderth, Hanspeter Pfister y Michael J Black. «A fully-connected layered model of foreground and background flow». En: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2013, págs. 2451-2458.

[19] Álvaro Artola Moreno. «Clasificación de imágenes usando redes neuronales convolucionales en Python». En: (2019).

[20] Ross Girshick, Jeff Donahue, Trevor Darrell y Jitendra Malik. «Rich feature hierarchies for accurate object detection and semantic segmentation». En: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, págs. 580-587.

[21] Ross Girshick. «Fast r-cnn». En: *Proceedings of the IEEE international conference on computer vision*. 2015, págs. 1440-1448.

[22] Rachel Huang, Jonathan Pedoeem y Cuixian Chen. «YOLO-LITE: a real-time object detection algorithm optimized for non-GPU computers». En: *2018 IEEE International Conference on Big Data (Big Data)*. IEEE. 2018, págs. 2503-2510.

[23] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu y Alexander C Berg. «Ssd: Single shot multibox detector». En: *European conference on computer vision*. Springer. 2016, págs. 21-37.

[24] Ugur Kart, Joni-Kristian Kämäräinen, Lixin Fan y Moncef Gabbouj. «Evaluation of Visual Object Trackers on Equirectangular Panorama.» En: *VISIGRAPP (5: VISAPP)*. 2018, págs. 25-32.

[25] Benjamin Coors, Alexandru Paul Condurache y Andreas Geiger. «Spherenet: Learning spherical representations for detection and classification in omnidirectional images». En: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, págs. 518-533.

[26] Chiyu Jiang, Jingwei Huang, Karthik Kashinath, Philip Marcus, Matthias Niessner y col. «Spherical CNNs on unstructured grids». En: *arXiv preprint arXiv:1901.02039* (2019).

[27] Kaiming He, Xiangyu Zhang, Shaoqing Ren y Jian Sun. «Deep residual learning for image recognition». En: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, págs. 770-778.

[28] Banş Evrim Demiröz, Ismail Ari, Orhan Eroğlu, Albert Ali Salah y Laie Akarun. «Feature-based tracking on a multi-omnidirectional camera dataset». En: *2012 5th International Symposium on Communications, Control and Signal Processing*. IEEE. 2012, págs. 1-5.

[29] Peng Chen, Xiang Chen, Beibei Jin y Xiangbing Zhu. «Online EM algorithm for background subtraction». En: *Procedia Engineering* 29 (2012), págs. 164-169.

[30] Brendan Juba y Hai S Le. «Precision-recall versus accuracy and the role of large data sets». En: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. 01. 2019, págs. 4039-4048.

[31] David Caruso, Jakob Engel y Daniel Cremers. «Large-scale direct slam for omnidirectional cameras». En: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2015, págs. 141-148.

[32] Nikolaos Zioulis, Antonis Karakottas, Dimitrios Zarpalas y Petros Daras. «Omnidepth: Dense depth estimation for indoors spherical panoramas». En: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, págs. 448-465.